NASA Contractor Report 172256

NASA-CR-172256
19840008427

Progress on a Generalized Coordinates Tensor Product Finite Element 3DPNS Algorithm for Subsonic Flow

A. J. Baker and J. A. Orzechowski

Computational Mechanics Consultants, Inc.
Knoxville, TN 37920

Contract NAS1 - 15105
December 1983

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

TABLE OF CONTENTS

SUMMARY

A generalized coordinates form of the penalty finite element algorithm for the three-dimensional parabolic Navier-Stokes equations for turbulent subsonic flows has been derived. This algorithm formulation requires only three distinct hypermatrices, and is applicable using any boundary fitted coordinate transformation procedure. The tensor matrix product approximation to the Jacobian of the Newton linear algebra matrix statement has been derived. The Newton algorithm has been restructured to replace large sparse matrix solution procedures with grid sweeping, using α-block tridiagonal matrices, where α equals the number of dependent variables. Numerical experiments have been conducted, and the resultant data gives guidance on potentially preferred tensor product constructions for the penalty finite element 3DPNS algorithm.

INTRODUCTION

The time-averaged, three dimensional Navier-Stokes equations, for steady, turbulent, subsonic flow of a compressible, heat conducting fluid, can be simplified to admit use of an efficient space marching numerical solution procedure under certain restrictions. Baker, et.al. [1] documents the derivation of the simplified, so-called "parabolic Navier-Stokes" equation set using formal ordering arguments. References [2-7] document application of a penalty, finite numerical solution algorithm for the parabolic Navier-Stokes equations for a variety of subsonic flow configurations, including an embedding within an interaction algorithm to impose axial pressure gradient feedback. These results have provided a basic assessment of the accuracy, convergence and versatility aspects of the finite element penalty algorithm, as well as detailed comparison between experimental data and prediction for various turbulent flow geometries.

These numerical predictions have been accomplished using the CMC:3DPNS computer code [8-10] which has evolved and matured principally under support of this contract. With the theoretical construction well verified, the requirement to improve efficiency becomes of next importance. The specific goals of this contract modification were to construct and evaluate a matrix tensor product factorization of the Newton iteration algorithm and to investigate inclusion of an embedded Poisson equation solution procedure. The results of pertinent analyses are documented in this report.

SYMBOLS

a	boundary condition coefficient; parameter
A	initial value matrix; hypermatrix prefix
b	constant
B	hypermatrix prefix; matrix
C	turbulence model coefficient
Eu	Euler Number
f	function of known argument
F	finite element matrix; discretized equation system
h	metric coefficient
H	stagnation enthalpy; Hilbert space
i	index
j	index
J	Jacobian
k	finite element basis degree; turbulence kinetic energy
ℓ	summation index; differential operator
L	differential operator
M	number of finite elements spanning R^n
n	unit normal vector; dimension of space
N	finite element cardinal basis; discrete index
p	pressure; iteration index
q	heat flux vector; generalized dependent variable
Q	generalized semi-discrete dependent variable
R^n	spatial domain of differential operator
Re	Reynolds Number
s,S	source term
S_e	finite element assembly operator
u_i	velocity vector
$\overline{u_i' u_j'}$	Reynolds kinematic stress tensor
U	convection matrix
x_i	Cartesian coordinate system
∂	partial derivative operator
∂R	boundary of solution domain R^n
δ	Kronecker delta; parameter
δQ	iteration vector

Δ	mesh measure; increment
ε	isotropic dissipation function; parameter
ζ	transformed coordinate
η_i	curvilinear coordinate system
κ	heat conductivity coefficient
λ	multiplier
ν	kinematic viscosity
ρ	density
σ_{ij}	Stokes stress tensor
Σ	summation
ϕ	constraint dependent variable
ω	sublayer damping function
Ω	solution domain

Superscripts

e	finite element reference
h	solution approximation
p	iteration index
T	matrix transpose
$'$	ordinary derivative

Subscripts

α	dependent variable index
e	finite element reference
i,j,k,ℓ	tensor indices
j	integration step index
o	reference state

Notation

{ }	column matrix
[ ]	square matrix
$\cup$	union
$\in$	belongs to

PROBLEM STATEMENT

Overview

The basic requirements of a numerical algorithm construction for the three-dimensional parabolic Navier-Stokes (3DPNS) equation set are accuracy, geometric versatility and efficiency. Any theoretical basis, e.g., finite difference, finite volume, finite element, etc., applied to constructing a 3DPNS algorithm ultimately yields the linear algebra statement $\{F\} = \{0\}$, where elements of $\{F\}$ are strongly nonlinear functions of the dependent variable set $q_\alpha(x_j)$. Dependent upon the algorithm designer's decisions, members of the set q_α can include density, mean velocity vector, stagnation enthalapy, turbulent kinetic energy, isotropic dissipation function, pressure and/or scalar potential fields and six components of the Reynolds stress tensor.

For the subject finite element penalty algorithm statement, and the 3DPNS equation set, the functional form of the algebra statement is,

$$\{FI(k, \lambda, \theta, \Delta x_1, \{QI\})\} = \{0\} \tag{1}$$

In equation 1, k is the polynomial degree of the finite element basis selected to construct the semi-discrete approximation $q_\alpha^h(x_j)$ to $q_\alpha(x_j)$, λ is the penalty function scalar multiplier, θ is the implicitness factor of the downstream integration step Δx_1, where $\theta \equiv \frac{1}{2}$ is the trapezoidal rule, and $\{QI(x_1)\}$ is the array of expansion coefficients of $q_\alpha^h(x_j)$, evaluated at the node coordinates of the (spatial) discretization UR_e^2 of the 3DPNS solution domain $\Omega \equiv R^2 x\ x_1$. For all $\theta > 0$, equation 1 is a nonlinear algebraic equation system eligible for solution using any of a multitude of (approximation) procedures. These all can be interpreted within the framework of the basic Newton iteration algorithm matrix solution statement.

$$[J(\{QI\})]^p_{j+1} \{\delta QI\}^{p+1}_{j+1} = - \{FI\}^p_{j+1} \tag{2}$$

where p is the iteration index at step x_{j+1}, and

$$\{QI\}^{p+1}_{j+1} \equiv \{QI\}^p_{j+1} + \{\delta QI\}^{p+1}_{j+1} \tag{3}$$

The Jacobian [J] appearing in equation 2, is by definition the square matrix,

$$[J(\{QI\})] \equiv \frac{\partial\{FI\}}{\partial\{QJ\}} \tag{4}$$

where both I and J range $1 \rightarrow 16$.

This contractual project phase initiated evaluation of decisions made in construction of equations 1-4, as they principally affect algorithm accuracy and efficiency.

Parabolic Navier-Stokes Equations

The three-dimensional parabolic Navier-Stokes (3DPNS) equations are a simplification of the steady, three-dimensional time-averaged Navier-Stokes equations, which in Cartesian tensor conservation form are

$$L(\rho) = \frac{\partial}{\partial x_j}\left[\rho u_j\right] = 0 \tag{5}$$

$$L(\rho u_i) = \frac{\partial}{\partial x_j}\left[\rho u_i u_j + p\delta_{ij} + \rho\overline{u_i' u_j'} - \sigma_{ij}\right] = 0 \tag{6}$$

$$L(\rho H) = \frac{\partial}{\partial x_j}\left[\rho H u_j - u_i\sigma_{ij} + \rho\overline{H' u_j'} - \overline{u_i'\sigma_{ij}'} + q_j\right] = 0 \tag{7}$$

$$L(\rho k) = \frac{\partial}{\partial x_j}\left[\rho u_j k + (C_{k\varepsilon}\frac{k}{\varepsilon}\rho\overline{u_i' u_j'} - \mu\delta_{ij})\frac{\partial k}{\partial x_i}\right] + \rho\overline{u_i' u_j'}\frac{\partial u_i}{\partial x_j} + \rho\varepsilon = 0 \tag{8}$$

$$L(\rho\varepsilon) = \frac{\partial}{\partial x_j}\left[\rho u_j\varepsilon + C_{\varepsilon\varepsilon}\frac{k}{\varepsilon}\rho\overline{u_i' u_j'}\frac{\partial\varepsilon}{\partial x_i}\right] + C_\varepsilon^1\rho\overline{u_i' u_j'}\frac{\varepsilon}{k}\frac{\partial u_i}{\partial x_j} + C_\varepsilon^2\frac{\rho\varepsilon^2}{k} = 0 \tag{9}$$

In equations 5-9 the usual superscript bar notation denoting time-averaged quantities [11] has been deleted for clarity. The time-averaged dependent variables are density (ρ), mean momentum vector (ρu_i), pressure (p) and stagnation enthalpy (H). Further, δ_{ij} denotes the Kronecker delta, and the Stoke's stress tensor (σ_{ij}) and heat flux vector (q_i) are defined as,

$$\sigma_{ij} \equiv \frac{\mu}{Re}\left[E_{ij} - \frac{2}{3}\delta_{ij}\, E_{kk}\right] \tag{10}$$

$$q_j \equiv -\kappa \frac{\partial H}{\partial x_j} \tag{11}$$

where μ and κ are laminar viscosity and heat conductivity respectively. E_{ij} is the symmetric mean flow strain rate tensor.

$$E_{ij} = \left[\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right] \tag{12}$$

Finally, $-\overline{u_i u_j}$ is the symmetric Reynolds stress tensor with trace equal to 2k, where k is the turbulent kinetic energy. For present purposes, $\overline{u_i' u_j'}$ is assumed correlated in terms of k, u_j and ε, the isotropic dissipation function, in the form,

$$\overline{u_i' u_j'} \equiv C_i k \delta_{ij} - C_4 \frac{k^2}{\varepsilon} E_{ij} - C_2 C_4 \frac{k^3}{\varepsilon^2} E_{ik} E_{kj} \tag{13}$$

where the C_α, $1 \leq \alpha \leq 4$, are known constants [1].

The parabolic approximation to the steady flow Navier-Stokes set, equations 5-13, is generated by assuming a principal direction of the flow persists, say parallel to the (curvi-linear) coordinate x_1. Assuming the corresponding mean velocity component u_1 is of order unity, i.e., O(1), and that the other two orthogonal components u_ℓ are smaller, say of $O(\delta)$, then for modest density variation the continuity equation 1 confirms that for $\partial/\partial x_1 \equiv O(1)$, then $\partial/\partial x_\ell = O(\delta^{-1})$. Proceeding through the analysis details [12] confirms that the O(1) 3DPNS equation set is

$$L(\rho) = \frac{\partial}{\partial x_j}\left[\rho u_j\right] = 0 \tag{14}$$

$$L(\rho u_1) = \frac{\partial}{\partial x_j}\left[\rho u_1 u_j + p\delta_{ij}\right] + \frac{\partial}{\partial x_\ell}\left[\rho\overline{u_1' u_\ell'} - \sigma_{1\ell}\right] = 0 \tag{15}$$

$$L(\rho u_\ell) = \frac{\partial}{\partial x_k}\left[p\delta_{k\ell} + \rho\overline{u_k' u_\ell'}\right] = 0 \tag{16}$$

$$L(\rho H) = \frac{\partial}{\partial x_j}\left[\rho H u_j\right] + \frac{\partial}{\partial x_\ell}\left[\rho\overline{H' u_\ell'} - \kappa\frac{\partial H}{\partial x_\ell} - u_i\sigma_{i\ell} - \overline{u_i'\sigma_{i\ell}'}\right] = 0 \tag{17}$$

$$L(\rho k) = \frac{\partial}{\partial x_j}\left[\rho k u_j\right] + \frac{\partial}{\partial x_\ell}\left[\left(\rho C_k \frac{k}{\varepsilon}\overline{u_j' u_\ell'} - \mu\delta_{j\ell}\right)\frac{\partial k}{\partial x_j}\right]$$

$$+ \rho\overline{u_i' u_\ell'}\frac{\partial u_1}{\partial x_\ell} + \rho\varepsilon = 0 \tag{18}$$

$$L(\rho\varepsilon) = \frac{\partial}{\partial x_j}\left[\rho\varepsilon u_j\right] + \frac{\partial}{\partial x_\ell}\left[\rho C_\varepsilon \frac{k}{\varepsilon}\overline{u_j' u_\ell'}\frac{\partial\varepsilon}{\partial x_j}\right] + C_\varepsilon^1\rho\overline{u_i' u_\ell'}\frac{\varepsilon}{k}\frac{\partial u_1}{\partial x_\ell}$$

$$+ C_\varepsilon^2\rho\varepsilon^2/k = 0 \tag{19}$$

In equations 14-19 the tensor index summation convention is $1 \leq (i,j) \leq 3$ and $2 \leq (k, \ell) \leq 3$. The Reynolds stress tensor constitutive equation 13 also becomes considerably simplified under the ordering analysis. For example, in rectangular Cartesian coordinates, and retaining the first two orders of terms yields equation 20. An equation of state $\rho = \rho(p,H)$ completes the basic 3DPNS statement.

$$
\begin{aligned}
& \overbrace{\qquad\qquad\qquad\qquad}^{O(\delta)} \qquad\qquad \overbrace{\qquad\qquad\qquad}^{O(\delta^2)} \\
\overline{u_1'u_1'} &= C_1 k - C_2 C_4 \frac{k^3}{\varepsilon^2}\left[\left(\frac{\partial u_1}{\partial x_2}\right)^2 + \left(\frac{\partial u_1}{\partial x_3}\right)^2\right] - 2C_4\frac{k^2}{\varepsilon}\left[\frac{\partial u_1}{\partial x_1}\right] \\
\overline{u_2'u_2'} &= C_3 k - C_2 C_4 \frac{k^3}{\varepsilon^2}\left[\frac{\partial u_1}{\partial x_2}\right]^2 - 2C_4\frac{k^2}{\varepsilon}\left[\frac{\partial u_2}{\partial x_2}\right] \\
\overline{u_3'u_3'} &= C_3 k - C_2 C_4 \frac{k^3}{\varepsilon^2}\left[\frac{\partial u_1}{\partial x_3}\right]^2 - 2C_4\frac{k^2}{\varepsilon}\left[\frac{\partial u_3}{\partial x_3}\right] \\
\overline{u_1'u_2'} &= - C_4\frac{k^2}{\varepsilon}\left[\frac{\partial u_1}{\partial x_2}\right] - C_2 C_4 \frac{k^3}{\varepsilon^2}\left[\frac{\partial u_1}{\partial x_3}\left(\frac{\partial u_2}{\partial x_3} + \frac{\partial u_3}{\partial x_2}\right)\right. \\
&\qquad\qquad \left. + 2\frac{\partial u_1}{\partial x_2}\left(\frac{\partial u_1}{\partial x_1} + \frac{\partial u_2}{\partial x_2}\right)\right] \\
\overline{u_1'u_3'} &= - C_4\frac{k^2}{\varepsilon}\left[\frac{\partial u_1}{\partial x_3}\right] - C_2 C_4 \frac{k^3}{\varepsilon^2}\left[\frac{\partial u_1}{\partial x_2}\left(\frac{\partial u_2}{\partial x_3} + \frac{\partial u_3}{\partial x_2}\right)\right. \\
&\qquad\qquad \left. + 2\frac{\partial \tilde{u}_1}{\partial x_3}\left(\frac{\partial u_1}{\partial x_1} + \frac{\partial u_3}{\partial x_3}\right)\right] \\
\overline{u_2'u_3'} &= - C_2 C_4 \frac{k^3}{\varepsilon^2}\left[\frac{\partial u_1}{\partial x_2}\frac{\partial u_1}{\partial x_3}\right] - C_4\frac{k^2}{\varepsilon}\left[\frac{\partial u_2}{\partial x_3} + \frac{\partial u_3}{\partial x_2}\right]
\end{aligned}
\tag{20}
$$

3DPNS Equation Set Completion

Equations 14-20 do not represent a well-posed, initial-boundary value problem statement for the dependent variable set. In particular, equation 14 is the sole definition for u_ℓ, yielding an underdetermined system. The finite element penalty algorithm construction of Baker [1] yields a well-posed problem statement by inclusion of both $O(\delta)$ transverse momentum equations, definition of an auxiliary harmonic function for a penalty constraint, and definition of complementary and particular solutions to a pressure Poisson equation formed from equation 16.

The derived pressure Poisson equation is,

$$
L(p) \equiv \frac{\partial L(\rho u_\ell)}{\partial x_\ell} = \frac{\partial^2 p}{\partial x_\ell^2} + \frac{\partial^2}{\partial x_\ell \partial x_k}\left[\overline{\rho u_k' u_\ell'}\right] = 0 \tag{21}
$$

the solution to which is defined as

$$p(x_i) \equiv p_c(x_\ell, x_1) + p_p(x_\ell, x_1) \tag{22}$$

The complementary pressure $p_c(x_\ell, x_1)$ is the solution to the homogeneous form of equation 21 with exterior flow boundary conditions [1]. The particular pressure is computed throughout the 3DPNS solution from equation 21, and added in a delayed manner into equation 22, used in the u_1 momentum equation solution, yielding a multi-pass interaction algorithm.

The retained $O(\delta)$ transverse momentum equation set is

$$L^{\delta}(\rho u_\ell) = \frac{\partial}{\partial x_j}\left[\rho u_\ell u_j\right] - \frac{\partial}{\partial x_k}\left[\sigma_{k\ell}\right] = 0 \tag{23}$$

The auxiliary harmonic variable ϕ is defined as the solution to a Poisson equation, driven by the continuity equation 13, in the form

$$L(\phi) \equiv \frac{\partial^2 \phi}{\partial x_\ell^2} - \frac{\partial}{\partial x_j}\left[\rho u_j\right] = 0 \tag{24}$$

Finite Element Penalty Algorithm

As a consequence of the 3DPNS equation set completion, equations 15, 16+23, 17+19, 21 and 24, define a well-posed, initial-boundary value statement for the dependent variable set $q_\alpha(x_j) \rightarrow \{q(x_j)\} = \{\rho u_1, \rho u_\ell, \rho H, \rho k, \rho \varepsilon, p_c, p_p, \phi\}$. The equation of state, $\rho = \rho(p,H)$, and equation 20 are algebraic definitions for the remaining seven members $\{\rho, \overline{u_i' u_j'}\}$. Therefore, the first nine members of $\{q(x_j)\}$ are eligible for constraint on the solution domain boundary, $\partial\Omega \equiv \partial R \times x_1$, by a linear combination of Dirichlet and Neumann boundary conditions of the form

$$\ell(q_\alpha) \equiv a_1^\alpha q_\alpha + a_2^\alpha \frac{\partial q_\alpha}{\partial x_\ell} \hat{n}_\ell + a_3^\alpha = 0 \tag{25}$$

In equation 25, the a_j^i are defined to enforce the appropriate constraint for each variable, c.f., Baker [2, 12]. Since the remaining seven members of $\{q(x_j)\}$ are defined by algebraic equations, no boundary conditions are appropriate. Finally, the first six members of $\{q(x_j)\}$ are required defined on the initial solution plane, $\Omega_o \equiv R^2 \times x_1(0)$, by an appropriate initial condition, $\{q_o(x_\ell, x_1(0))\}$.

The complete derivation of the finite element penalty constraint numerical solution algorithm is given in [1, 12]. Briefly, the semi-discrete approximation for each member of the set $q_\alpha(x_j)$ is formed by the union of elemental approximations $q^e(x_j)$ as,

$$q_\alpha(x_j) \approx q_\alpha^h(x_j) \equiv \bigcup_e q_\alpha^e(x_j) \tag{26}$$

In turn, each elemental semi-discrete approximation, valid on the representative finite element domain $\Omega_e \equiv R_e^2 \times x_1$, is formed as an expansion on the cardinal basis $\{N_k(x_\ell)\}$, the members of which are (typically) polynominals complete to degree k, in the form

$$q_\alpha^e(x_j) \equiv \{N_k(x_\ell)\}^T \{QI(x_1)\}_e \tag{27}$$

In equation 27, $\{\cdot\}$ denotes a column matrix, superscript T its transpose, subscript e denotes pertaining to R_e^2 , and the elements of $\{QI\}_e$ are the evaluation of the semi-discrete approximation at the nodal coordinates of R_e^2.

A basic requirement in any algorithm construction is a formal statement regarding constraint on the error formed by employing the semi-discrete approximation for the differential equation set. The finite element algorithm construction requires the semi-discrete approximation error to be orthogonal to the basis employed to construct $q_\alpha^h(x_j)$. For all members of $\{q^h\}$ except u_ℓ^h , the resultant error constraint statement is,

$$\int_{R^2} \{N_k(x_\ell)\}\, L(q_\alpha^h)d\vec{x} + \beta \int_{\partial R} \{N_k(x_\ell)\}\ell(q_{\alpha}^h)dx \equiv \{0\} \tag{28}$$

where β is a scalar multiplier selected to achieve cancellation of the middle term in equation 25. The error extremization statement for the members u_ℓ^h of $\{q^h\}$ is,

$$\int_{R^2} \{N_k(x_\ell)\} \left[L(u_\ell^h) + L^\delta(u_\ell^h)\right] d\vec{x} + \beta \int_{\partial R} \{N_k(x_\ell)\} \ell(u_\ell^h) dx$$

$$+ \lambda \int_{R^2} \frac{\partial \{N_k\}}{\partial x_\ell} L(\phi^h) d\vec{x} \equiv \{0\} \tag{29}$$

where λ is an arbitrary parameter modifying the penalty term which constrains the error extremization by the continuity equation (error).

Equations 28-29 define the finite element penalty algorithm semi-discrete error constraint statement for the 3DPNS equation set. For the non-initial-valued dependent variables, equation 28 yields the linear algebra statement $\{FI\} = \{0\}$, recall equation 1. For the initial-valued variables, equations 28-29 yield a coupled ordinary differential equation set,

$$[A] \frac{d\{QI\}}{dx_1} + \{B(QI)\} = \{0\} \tag{30}$$

which is transformed to a linear algebra statement using a Taylor series, for example,

$$\{FI\} \equiv \{QI\}_{j+1} - \{QI\}_j - \Delta x_1 \{QI\}'_{j+\theta} + \cdots = \{0\} \tag{31}$$

where superscript prime denotes the ordinary derivative and $\theta > 0$ implies an implicit statement since equation 30 is quite nonlinear.

Hence, the final fully discrete approximation error constraint statement is the nonlinear algebraic equation set,

$$\{FI(k, \lambda, \theta, \Delta x_1, \{QI\})\} = \{0\} \tag{32}$$

where $1 \leq I \leq 16$, see equation 1. The Newton algorithm solution for equation 32 is given in equations 2-4.

DISCUSSION AND RESULTS

Generalized Coordinates

A basic requirement for a 3DPNS algorithm is geometric versatility, such that the discretization of the transverse plane solution domain R^2 can be efficiently embedded within a boundary comprised of the union of select aerodynamic surfaces and freestream interfaces. The term "generalized coordinates" has gained acceptance in describing an algorithm construction suitable for use with a regularizing, boundary fitted coordinate transformation. This construction also impacts directly on the efficiency of an implicit algorithm, since it usually facilitates factorization of the linear algebra statement Jacobian, recall equation 2. An objective of the current project is to construct and evaluate a matrix tensor product approximation to the Newton Jacobian of the finite element 3DPNS algorithm [8].

The required step is to derive the generalized coordinates form of the finite element penalty algorithm, equations 28-29. A multitude of procedures are available to generate regularizing transformations, c.f., [13], and each may be viewed as generating an approximation to the mapping,

$$x_i = x_i(\eta_j) \tag{33}$$

at a finite number of coordinate triples on the domain $\Omega = R^2 \times x_1$. For the space-marched 3DPNS equation set, equation 33 may be conveniently decomposed into an x_1-oriented grid-stretching transformation, $x_1 = x_1(\xi_j)$, and a regularizing transformation on R^2 mapping the boundaries ∂R onto coordinate surfaces of η_j, $j = 1, 2$. Denote the generated set of coordinate pairs on R^2 as $\{XI\}$, $I = 2, 3$. The number of entries in $\{XI\}$ equals twice the mesh characterization of the discretization UR^2_e of R^2. (For example, for a 41X41 mesh, there are $2(41)(41) = 3362$ entries in $\{XI\}$.) On any (each) finite element domain R^2_e, the specific form of equation 33 is,

$$x_i = \{N_k(\vec{\eta})\}^T \{XI\}_e \tag{34}$$

for $x_i \in R^2_e$. In equation 34, the elements of $\{XI\}_e$ are the appropriate members of $\{XI\}$, and the interpolation basis $\{N_k(\vec{\eta})\}$ is (potentially) identical to the approximation subspace for q^h_i, equation 27.

Equation 34 is of general utility since the elements of $\{N_k(\vec{n})$ are well known for $k \geq 1$, using either triangular or quadrilateral shaped finite elements R_e^2, with or without curved sides. The algorithm requirement is to transform the derivatives $\partial/\partial x_1$ and $\partial/\partial x_\ell$, see equations 15-19, 21-24, as they appear in the penalty algorithm statement, equations 28-29. A grid stretching parallel to the x_1 coordinate direction introduces additional derivatives in the x_ℓ plane, upon transformation to the ζ_j coordinate system of the form [8].

$$\frac{\partial}{\partial x_1} \rightarrow \frac{\partial}{\partial \zeta} - h_2 \frac{\partial}{\partial x_2} - h_3 \frac{\partial}{\partial x_3} = \frac{\partial}{\partial \zeta} - h_\ell \frac{\partial}{\partial x_\ell} \tag{35}$$

Therefore, the general form of the initial-valued partial differential equations 15, 16+23, 17-19, of the 3DPNS set is

$$L(q_\alpha) = \frac{\partial}{\partial \zeta}\left[u_1 q_\alpha\right] + \frac{\partial}{\partial x_\ell}\left[(1-h_\ell) q_\alpha u_\ell + \tau_{\alpha\ell}\right] + s_\alpha = 0 \tag{36}$$

Table 1 lists q_α, $\tau_{\alpha\ell}$, and s_α for $1 \leq \alpha \leq 6$ in equation 36, and $2 \leq \ell \leq 3$.

Table 1. Variables and Parameters in Equation 36

α	q_α	$\tau_{\alpha\ell}$	s_α
1	ρu_1	$\overline{\rho u_1 u_\ell} - \sigma_{1\ell} - h_\ell p$	$\frac{\partial p}{\partial \zeta}$
2	ρu_2	$\overline{\rho u_2 u_\ell} - \sigma_{2\ell} + p\delta_{2\ell}$	0
3	ρu_3	$\overline{\rho u_3 u_\ell} - \sigma_{3\ell} + p\delta_{3\ell}$	0
4	ρH	$\overline{\rho H u_\ell} - \kappa \frac{\partial H}{\partial x_\ell}$	$-\frac{\partial}{\partial x_\ell}\left[u_i \sigma_{i\ell} + \overline{u_i \sigma_{i\ell}}\right]$
5	ρk	$\left(C_k \frac{k}{\varepsilon} \overline{\rho u_j u_\ell} - \mu\delta_{j\ell}\right) \frac{\partial k}{\partial x_j}$	$\overline{\rho u_\ell u_\ell} \frac{\partial u_1}{\partial x_\ell} + \rho\varepsilon$
6	$\rho\varepsilon$	$C_\varepsilon \frac{k}{\varepsilon} \overline{\rho u_j u_\ell} \frac{\partial \varepsilon}{\partial x_j}$	$C_\varepsilon^1 \overline{\rho u_1 u_\ell} \frac{\varepsilon}{k} \frac{\partial u_1}{\partial x_\ell} + C_\varepsilon^2 \frac{\rho\varepsilon^2}{k}$

The generalized coordinates construction requires transformation of the planar divergence operator $\partial/\partial x_\ell$, when equation 36 is inserted into equation 28. Using a Green-Gauss form of the divergence theorem on the second term in equation 36, yields

$$\int_{R^2} \{N_k\}\, L\left(q_\alpha^h\right) d\vec{x} \;+\; \beta \int_{\partial R} \{N_k\} \ell(q_\alpha^h) d\sigma = \int_{R^2} \{N\} \left[\frac{\partial}{\partial \xi}\left(u_1^h q_\alpha^h\right) + s_\alpha^h\right] d\vec{x}$$

$$+ \oint_{\partial R} \{N\}\left[(1-h_{\underline{\ell}}) q_\alpha^h u_\ell^h + \tau_{\alpha\ell}^h\right] \hat{n}_\ell d\sigma$$

$$- \int_{R^2} \frac{\partial\{N\}}{\partial \eta_k} \frac{\partial \eta_k}{\partial x_\ell} \left[\left(1-h_{\underline{\ell}}\right) q_\alpha^h u_\ell^h + \tau_{\alpha\ell}^h\right] d\vec{x}$$

$$+ \beta \int_{\partial R} \{N\}\left[a_1^\alpha\, q_\alpha^h + a_2^\alpha \frac{\partial q_\alpha^h}{\partial \eta_k} \frac{\partial \eta_k}{\partial x_\ell}\, \hat{n}_\ell + a_3^\alpha\right] d\sigma \qquad (37)$$

The evaluation of equation 37 is accomplished by performing the integrals on an element-by-element basis, and assembling the resultant contributions into {FI} using the matrix assembly operator $S_{\underline{e}}$, c.f., [12, Ch. 2]. Since the elements of $\{N_k(\vec{\eta})\}$ are known functions of $\vec{\eta}$, the only requirement is evaluation of $\partial\eta_k/\partial x_\ell$ on a generic finite element domain $R^2_{\underline{e}}$. From equation 34, the 2X2 square matrix defining the Jacobian of the forward transformation is

$$[J]_e = [J(\{XI\}_e)] \equiv \left[\frac{\partial x_\ell}{\partial \eta_k}\right]_e \tag{38}$$

Thus, the elements of the inverse transformation Jacobian are,

$$\left[\frac{\partial \eta_k}{\partial x_\ell}\right]_e = [J]_e^{-1} = \frac{1}{\det J_e}[C]_e \tag{39}$$

where [C] is the 2X2 transformed cofactor matrix of $[J]_e$, the elements of which are algebraic functions of η_k and $\{XI\}_e$. The differential element $d\vec{x}$ in equation 37 becomes

$$d\vec{x} = \det J_e \, d\vec{\eta} \tag{40}$$

Finally, it is convenient to define the contravariant components of the convection velocity semi-discrete approximation $\bar{u}_k^e$ on R_e^2 as

$$\bar{u}_k^e \equiv \det J_e \left[\frac{\partial \eta_k}{\partial x_\ell}\right]_e (1-h_\ell)u_\ell^e = (1-h_\ell)\,[C]_{k\ell}^e\, u_\ell^e \tag{41}$$

With equations 38-41, and recasting evaluation of equation 37 as the assembly of integrals over UR_e^2, the generalized coordinates form of the finite element algorithm statement, equation 28, becomes

$$\int_{R_2} \{N_k(\vec{\eta})\}\, L\left(q_\alpha^h\right) d\vec{x} + \beta \int_{\partial R} \{N_\kappa(\vec{\eta})\}\ell\left(q_\alpha^h\right) d\sigma$$

$$= S_e \left[\int_{R_e^2} \det J_e \, \{N\}\,\{N\}^T \left[\frac{d}{d\zeta}\{U1QI\}_e + \{SI\}_e\right] d\vec{\eta}\right.$$

$$- \int_{R^2} \{UBARK\}_e^T \{N\} \frac{\partial}{\partial \eta_k} \{N\} \{N\}^T \{QI\}_e \, d\vec{\eta}$$

$$- \int_{R_e^2} \{ETAKL\}_e^T \{N\} \frac{\partial}{\partial \eta_k} \{N\} \{N\}^T \{TAUIL\}_e \, d\vec{\eta}$$

$$+ \kappa \int_{\partial R} \{N\} \left[a_1^4 \, q_4^e + a_3^4 \right] \det J_e \, d\sigma$$

$$+ \oint_{\partial R} \{N\} \left[(1-h_{\underline{\ell}}) \{UL\}^T \{N\} \{N\}^T \{QI\}_e \right.$$

$$\left. + \{N\}^T \{TAUIL\}_e \hat{n}_L \det J_e \right] d\sigma \Bigg] \qquad (42)$$

In writing equation 42, it has been assumed the only variable possessing a constrained normal derivative boundary condition is stagnation enthalpy. Hence, the fourth term contains only q_4^e, and the normal derivative term in equation 25 is cancelled by the corresponsing term in the third integral in equation 38 by defining $\beta \equiv \kappa$, see Table 1. A second assumption is that $\det J_e$ is adequately represented as an elemental scalar, which is a commission of interpolation error (only) on a sufficiently refined mesh. The elements of the direction cosine matrix $\{ETAKI\}_e$ are defined by the interpolation.

$$\left[\frac{\partial \eta_\ell}{\partial x_k} \right] \equiv \frac{1}{\det J_e} \{N_k\}^T \{ETAKL\}_e \qquad (43)$$

For example, defining k = 1 in equation 43, for the bilinear tensor product basis on straight-sided quadrilaterals, numbering nodes counter clockwise, and defining the elements of the nodal array $\{XI\}_e$ as $\{YJ, ZJ, 1 \leq J \leq 4\}_e$, the four arrays $2\{ETAKL\}_e$ are, c.f., [12, Ch. 8].

$$2\{ETAKL\}_e = \left\{ \begin{matrix} Z4-Z1 \\ Z3-Z2 \\ Z3-Z2 \\ Z4-Z1 \\ (2,2) \end{matrix} \right\}_e \left\{ \begin{matrix} Y1-Y4 \\ Y2-Y3 \\ Y2-Y3 \\ Y1-Y4 \\ (2,3) \end{matrix} \right\}_e, \left\{ \begin{matrix} Z1-Z2 \\ Z1-Z2 \\ Z4-Z3 \\ Z4-Z3 \\ (3,2) \end{matrix} \right\}_e, \left\{ \begin{matrix} Y2-Y1 \\ Y2-Y1 \\ Y3-Y4 \\ Y3-Y4 \\ (3,3) \end{matrix} \right\}_e \tag{44}$$

The indices in the parenthesis in equation 44 denote (K, L). Finally, the matrix elements of $\{UBARK\}_e$ are defined by the interpolation.

$$\overset{-e}{U}_k \equiv \{N_k\}^T \{UBARK\}_e \tag{45}$$

Proceeding through the algebra yields

$$\{UBARK\}_e = \{ETAKL\}_e \{N_k\}^T \{UL\}_e \tag{46}$$

Since the evaluation of equation 46 is at nodes, the elements of $\{N_k\}$ reduce to the Kronecker delta; hence, $\{N_k\}^T\{UL\}_e$ simply selects the correct nodal value out of the elemental arrays $\{UL\}_e$, L = 2, 3.

In equations 42-46, the indices K, L are tensor summation indices, while I denotes the appropriate member of $\{q_k^h\}$. Every matrix denoted by a subscript e in equation 42 is independent of the $\vec{\eta}$ coordinate system spanning R_e^2, hence can be extracted outside the integral. The expressions remaining within the integrand are polynomials on $\vec{\eta}$, for all definitions of completeness k of the semi-discrete approximation, recall equations 26-27. The order of $\{N_k\}$ depends upon k, as do all the integrels in equation 42, and each is readily evaluated using numerical quadrature. The basic structure of equation 42 is thus defined, using a standardized hypermatrix nomenclature [12], as

$$\int_{R^2} \{N_k\}\, L\left(q_\alpha^h\right) d\vec{x} + \beta \int_{\partial R} \{N_k\}\, \ell\left(q_\alpha^h\right) d\sigma$$

$$= S_e \Bigg[\det J_e\, [B200]\, \frac{d}{d\zeta} \{U1QI\}_e + \det J_e\, [B200]\, \{SI\}_e$$

$$- \left(\{UBARK\}_e^T\, [B30K0] - \left(1-h_{\underline{k}}\right) \hat{n}_K \det J_e\, \{UK\}_e^T\, [A3000] \right) \{QI\}_e$$

$$- \left(\{ETAKL\}_e^T\, [B30K0] - \hat{n}_L \det J_e\, [A200]\, \{TAUIL\}_e \right.$$

$$\left. + \kappa\, \delta_{I4} \det J_e \left(a_1^4 [A200]\, \{QI\}_e + a_3^4\, \{A10\} \right) \right] \equiv \{0\} \qquad (47)$$

Note that equation 47 is in the form of equation 30, with the definition $[A] \equiv S_e[\det J_e[B200]]$. The generalized coordinates finite element penalty algorithm involves definition of only three distinct standard matrices, [B200] and [B30K0], K = 2, 3, on R_e^2. These are independent of R_e^2 and depend only on the degree k of the semi-discrete approximation. In addition, there are the three standard matrices [A200], [A3000] and {A10}, defined on the boundary of R^2, hence evaluated on a one-dimensional element R_e^1. In the second and third lines of equation 47, the terms involving [A ··] are the residuals from use of the Green-Gauss theorem, see equation 37. The terms in the fourth line result from the heat convection boundary condition for stagnation enthalpy. The tensor indices range $2 \leq (K, L) \leq 3$, the subscript $\underline{k}$ takes on the value of K, $\hat{n}_K$ and $\hat{n}_L$ are unit normal vectors to R^2, and $1 \leq I \leq 16$ denotes the ordered members of the dependent variable array. Finally, $\det J_e$ is the average value on R_e^2 of the determinant of the transformation Jacobian, equation 38, and $\{ETAKL\}_e$ and $\{UBARK\}_e$ are element dependent matrices defined in equations 43-46. The Appendix lists the defined matrices [A · ·] and [B· ·]. for k = 1 in equation 27.

Tensor Matrix Product Jacobian

The form of equation 47 is well suited to formation of a tensor (outer) matrix product approximation to the Newton iteration algorithm Jacobian, see equations 2 and 4, upon definition and use of a tensor product cardinal basis $\{N_k^+\}$, equation 34, on quadrilateral finite element domains R_e^2 (which could be curved-sided, $k > 1$). In this instance, the Newton algorithm Jacobian is approximated by the tensor matrix product construction,

$$[J(\{QI\})] \Rightarrow [J_2] \otimes [J_3] \tag{48}$$

where $\otimes$ denotes the outer product [14]. Equation 2 then becomes of the form

$$[J_2] \otimes [J_3] \{\delta QI\} = - \{FI\} \tag{49}$$

Making the definition $[J_3] \{\delta QI\} \equiv \{\delta PI\}$, equation 49 is solved by sweeping the mesh parallel to η_2 , to determine $\{\delta PI\}$, and then sweeping the mesh parallel to η_3 to complete the solution, i.e.,

$$\begin{aligned} [J_2] \{\delta PI\}_e &= - \{FI\}_e \\ [J_3] \{\delta QI\}_e &\equiv \{\delta PI\}_e \end{aligned} \tag{50}$$

The attraction of the approximate solution statement, equation 50, is replacement of the large sparse matrix [J], equation 4, by two block tri- or penta-diagonal matrices $[J_\eta]$, with the corresponding significant reduction in computer storage and LU decomposition computer CPU time. The detraction of the statement is degradation of the convergence rate associated with the exact Newton algorithm statement. Of course, for the 16-dependent variable 3DPNS equation set it is impossible to construct and use the exact Jacobian. Hence, the trade-off is degree of approximation versus associated cost. The critical measure is convergence rate since this determines the number of passes through the linear algebra statement.

Study Problem Statement

The tensor matrix product approximation to the 3DPNS Newton Algorithm Jacobian is formed by evaluation of the appropriate integrals in $\partial\{FI\}/\partial\{QJ\}$

on a one-dimensional domain R_e^1. The CMC:3DPNS code [10] was reorganized to permit construction of a one-dimensional Jacobian, independent of the subroutines used to generate {FI}. The basic issue is convergence, and the study problem is solution of the 2DPNS equations for laminar and/or turbulent flow. The code was indexed to permit use of the original algorithm formulation, as well as several variants of the tensor product construction. The 2DPNS algorithm construction for turbulent flow using the k-ε closure with a Reynolds stress algebraic model was established, however, the detailed discussion is limited to the equation system for laminar isoenergetic flow.

The governing 2DPNS differential equation system, see equation 36, for $\{q\} = \{u, v, p, \phi\}$, is

$$L(u) = \frac{\partial}{\partial x}(uu) + p' + \frac{\partial}{\partial y}\left(vu - \frac{\nu}{Re}\frac{\partial u}{\partial y}\right) = 0 \tag{51}$$

$$L^{\delta}(v) = \frac{\partial}{\partial x}(uv) + \frac{\partial}{\partial y}\left(vv + \frac{p}{\rho} - \frac{\nu}{Re}\frac{\partial v}{\partial y}\right) = 0 \tag{52}$$

$$L(p) = \frac{\partial}{\partial y}\left[\frac{1}{\rho}\frac{\partial p}{\partial y} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} - \frac{1}{Re}\frac{\partial}{\partial y}\left(\nu\frac{\partial v}{\partial y}\right)\right] = 0 \tag{53}$$

$$L(\phi) = \frac{\partial^2 \phi}{\partial y^2} - \frac{\partial u}{\partial x} - \frac{\partial v}{\partial y} = 0 \tag{54}$$

In equation 51, $p' \equiv \frac{1}{\rho}\frac{dp}{dx}$ is an input parameter (corresponding to the $p_c(x_1, x_\ell)$ solution, equations 21-22, for 3DPNS). Inserting equations 51-54 into the finite element penalty algorithm, equations 28-29, see equation 42, proceeding through the algebra, equation 47, and inserting the results for u^h and v^h into the trapezoidal rule ($\theta = \frac{1}{2}$) form of equation 31, equation 32 becomes the set.

$$\{FI\} \equiv \underset{e}{S}\{FU\}_e = \underset{e}{S}\left[\frac{1}{2}\Delta_e\{U^p_{j+1} + U_j\}^T_e [A3000]\left(\{U\}^p_{j+1} - \{U\}_j\right)\right.$$

$$\left. + \frac{\Delta x}{2}\left(\{V\}^T_e [A3001]\{U\}_e + \Delta_e\{P'\}_e + \frac{1}{\Delta_e}\{NU\}^T[A3011]\{U\}\right)\right]_{j+1,j} \tag{55}$$

$$\{F2\} \equiv \underset{e}{S} \{FV\}_e = \underset{e}{S}\Bigg[\frac{1}{2}\Delta_e\{U^p_{j+1} + U_j\}^T_e [A3000]\left(\{V\}^p_{j+1} - \{V\}_j\right)$$

$$+ \frac{\Delta x}{2}\Bigg[\{V\}^T_e [A3001]\{V\}_e + \frac{1}{\rho}[A201]\{P\}_e$$

$$+ \frac{1}{\Delta_e}\{NU\}^T_e [A3011]\{V\}_e\Bigg]_{j+1,\,j}$$

$$+ C_\phi\left[\lceil U\rfloor_e [A201]\{S\phi\}_e\right]_{j+1}\Bigg] \qquad (56)$$

$$\{F3\} \equiv \underset{e}{S}\{FP\}_e = \underset{e}{S}\Bigg[\frac{1}{\rho\Delta_e}[A211]\{P\}_e + \frac{1}{Eu}\{U\}^T_e [A3010]\{VP\}_e$$

$$+ \frac{1}{Eu\Delta_e}\{V\}^T_e [A3011]\{V\}_e\Bigg]_{j+1} \qquad (57)$$

$$\{F4\} \equiv \underset{e}{S}\{F\phi\}_e = \underset{e}{S}\Bigg[\frac{1}{\Delta_e}[A211]\{\phi\}_e + \Delta_e [A200]\{UP\}_e + [A201]\{V\}_e\Bigg]_{j+1} \qquad (58)$$

The newly defined standard matrices $[A\cdot\cdot]$ are also listed in the Appendix. Further, $\det J_e \equiv \Delta_e$, the measure of R^1_e, and its occurrence has been cleared throughout equations 55-58, such that $[A\cdots]$are integer arrays with a common divisor. The continuity equation has been explicitly employed to clear extra terms in equations 55-56, $\{\cdot\}^p_{j+1}$ denotes evaluation with the current iterate at x_{j+1}, $\{\cdot\}_j$ denotes the converged solution at the previous step (x_j), $(\cdot)_{j+1,j}$ indicates evaluation with both $\{\cdot\}^p_{j+1}$ and $\{\cdot\}_j$, followed by addition, and $[\cdot]_{j+1}$ denotes evaluation with $\{\cdot\}^p_{j+1}$. The arrays $\{UP\}_e$ and $\{VP\}_e$ contain a finite difference approximation for $\frac{d}{d\zeta}\{U\}_e$ and $\frac{d}{d\zeta}\{V\}_e$, evaluated at x_{j+1}. The elements of $\{NU\}_e$ are ν/Re, where Re is the Reynolds number, Eu is

the Euler number, and $\{P'\}_e$ contains $\frac{1}{\rho}\frac{dp}{dx}$.

The last term in equation 56 is the continuity constraint penalty term, the selected form of the λ- modified integral in equation 29. C_ϕ is a constant of order unity, and $[U]_e$ is a diagonal matrix with elements taken from $\{U\}_e$ in the same order. (This multiplier improves the resolution of $\{V\}$ in the lower reaches of the boundary layer. The penalty term dependent variable is $\{S\phi\}_e$, defined as the sum of the previous solutions to equation 58 at step x_{j+1}, i.e.,

$$\begin{aligned}
\{S\phi\}_{j+1}^{p+1} &\equiv \{S\phi\}_{j+1}^{p} + \{\phi\}_{j+1}^{p+1} \\
&= \{S\phi\}_{j+1}^{p} + \left(\{\phi\}_{j+1}^{p} + \{\delta\phi\}_{j+1}^{p+1}\right) \\
&= \{S\phi\}_{j+1}^{p-1} + \left(\{\phi\}_{j+1}^{p-1} + \{\delta\phi\}_{j+1}^{p}\right) \\
&\quad + \left(\{\phi\}_{j+1}^{p} + \{\delta\phi\}_{j+1}^{p+1}\right), \text{ etc.}
\end{aligned} \tag{59}$$

Furthermore, $\{S\phi\}_{j+1}^{1} \equiv \{S\phi\}_j$, which is the identification as well for the first estimate of the dependent variable set, i.e., $\{QI\}_{j+1}^{1} \equiv \{QI\}_j$. This definition of $\{S\phi\}$, as the penalty variable, is a considerable modification of the orginal algorithm construction in CMC:3DPNS, to permit solution for $\{\delta\phi\}$ within equation 2, rather than a separate Poisson solution for $\{\phi\}$.

Equations 55-59 constitute the linear algebra specification of the 2DPNS algorithm. The Newton algorithm Jacobian matrices are derived from the definition, equation 4, using equations 55-58. They are identical to the factor $[J_2]$ of the tensor product approximation for the 3DPNS statement,

equation 50. Recalling that $[J] \equiv S_e[J]_e$, the elemental components of the 4-block, tridiagonal Jacobian $[J_2]$ are:

$$[JUU]_e = \Delta_e\{U\}_e^T\,[A3000] + \frac{\Delta x}{2}\left[\{V\}_e^T\,[A3001] + \frac{1}{\Delta_e}\{NU\}_e^T\,[A3011]\right]$$

$$[JUV]_e = \frac{\Delta x}{2}\{U\}_e^T\,[A3100]$$

$$[JUP]_e = [0]$$

$$[JU\phi]_e = [0] \qquad (60)$$

$$[JVV]_e = \frac{1}{2}\Delta_e\,\{U_{j+1}^p + U_j\}_e^T\,[A3000]$$

$$+ \frac{\Delta x}{2}\left[\{V\}_e^T\left([A3001] + [A3100]\right) + \frac{1}{\Delta_e}\{NU\}_e^T\,[A3011]\right]$$

$$[JVU]_e = \frac{1}{2}\Delta_e\,\{V_{j+1}^p - V_j\}_e^T\,[A3000]$$

$$[JVP]_e = \frac{\Delta_e}{2\rho}\,[A201]$$

$$[JV\phi]_e = C_\phi\,[U]_e[A201] \qquad (61)$$

$$[JPP]_e = \frac{1}{\rho\Delta_e}\,[A211]$$

$$[JPU]_e = \frac{1}{Eu}\,\{VP\}_e^T\,[A3010]$$

$$[JPV]_e = \frac{a}{Eu}\,\{U\}_e^T\,[A3010] + \frac{1}{Eu\Delta_e}\,\{V\}_e^T\left([A3011] + [A3110]\right)$$

$$[JP\phi]_e = [0] \qquad (62)$$

$$[J\phi\phi]_e = \frac{1}{\Delta_e}\,[A211]$$

$$[J\phi U]_e = a\Delta_e\,[A200]$$

$$[J\phi V]_e = \quad [A201]$$

$$[J\phi P]_e = \quad [0] \tag{63}$$

In equations 60-63 all evaluations are made using $\{\cdot\}^p_{j+1}$ unless otherwise noted. The parameter "a" in equations 62-63 denotes the fraction of $\{QI\}^p_{j+1}$ used in the finite difference approximation for $\frac{d}{d\zeta}\{QI\}$ at x_{j+1}.

Numerical Results

Numerical experiments have been executed to evaluate the 2DPNS algorithm using various approximations to the Newton algorithm Jacobian, equations 60-63. The exact construction is

$$S_e \begin{bmatrix} [JUU]_e & [JUV]_e & 0 & 0 \\ [JVU]_e & [JVV]_e & [JVP]_e & [JV\phi]_e \\ [JPU]_e & [JPV]_e & [JPP]_e & 0 \\ [J\phi U]_e & [J\phi V]_e & 0 & [J\phi\phi]_e \end{bmatrix}^p_{j+1} \{\delta QI\}^{p+1}_{j+1} = -\{FI\}^p_{j+1} \tag{64}$$

For reference, both the 2D and 3D algorithm constructions currently operational in the CMC:3DPNS code approximate equation 64 by its diagonal entries only. This decouples the dependent variable set solution in the nondiagonal terms premultiplied by Δx, eauations 60-63, but retains the Δx-coupling through convection, pressure gradient and viscous-turbulent effects. Further, the original algorithm Poisson equation solutions are on $\{p\}$ and $\{\phi\}$, rather than $\{\delta P\}$ and $\{\delta\phi\}$, and both are computed using the most recent solutions $\{U\}^{p+1}_{j+1}$, $\{V\}^{p+1}_{j+1}$, etc., rather than $\{QI\}^p_{j+1}$, see equation 2.

The 2D algorithm construction for solution of $\{P\}$ and $\{\phi\}$ was rearranged and the code modified to permit PNS solutions using various approximations to [J]. The nearest equivalent to the original algorithm construction is,

$$\begin{bmatrix} [JUU] & & & \\ & [JVV] & & \\ & & [JPP] & \\ & & & [J\phi\phi] \end{bmatrix}_{j+1}^{p} \{\delta QI\}_{j+1}^{p+1} = - \{FI\}_{j+1}^{p} \qquad (65)$$

where $\{FP\}$ and $\{F\phi\}$ are now evaluated using $\{QI\}_{j+1}^{p}$. Defining the penalty variable as $\{S\phi\}$, see equations 56 and 59, reproduced the original algorithm solution data to within negligible differences. The (finally) selected laminar flow standard test case used a geometrically nonuniform, unstretched grid of M=56 elements spanning $3\delta_0$. The initialization of $\{U\}_0$ employed the Blasius solution, and the boundary layer thickness doubled over the integration length $\Delta x = 4$, achieved in nominally 20 nonuniform integration steps Δx at $Re = 3.2 \times 10^6$.

Table 2 summarizes data confirming the modified 2DPNS algorithm can generate an acceptably accurate solution on the M=56 mesh, with nodal coordinates as listed in column 1. Columns 2-4 compare the axial velocity profiles of the Blasius solution, the direct boundary layer (2DBL) solution using [JUU] and a direct trapezoidal rule integration of the continuity equation 14, and the basic 2DPNS algorithm. The matrix iteration convergence requirement was $|\delta U| < 10^{-5}$, and peculiarities in starting the 2DPNS solution were exactly mimicked in generating the 2DBL solution. Hence, both numerical solutions are displaced axially from the Blasius solution by a negligible increment. The agreement between the 2DBL and 2DPNS solutions is excellent in both u_1 and in u_2, see columns 5-6 of Table 2. The transverse velocity distribution is the sensitive measure,since it ranges over five digits of significance over δ (see arrows) and is required to implicitly approximate a vanishing normal derivative at $y/\delta = 0$.

With this confirmation, the principal interest is efficiency. Table 3 summarizes comparison data on convergence of the approximate Newton algorithm Jacobian constructions. All computations are done in single precision using a 32-bit word. Using the Blasius solution for $\{V(x,y)\}$ the 2DBL data of the second column indicates use of [JUU] yields approximate quadratic convergence to E(-7) for a scalar equation. Appending the direct continuity equation solution

Table 2

Accuracy Comparisons, Laminar Boundary Layer Test Case, $\Delta X = 4.$

NODES	AXIAL VELOCITY - u_1 X 10^{-1}			TRANSVERSE VELOCITY - u_2 X 10^3	
$2y/10\delta_o$	Blasius	2DBL	2DPNS	2DBL	2DPNS
07999	099968	099965	099966	215451	216400
07777	099950	099945	099946	215312	216260
07555	099923	099916	099917	215106	216054
07333	099883	099872	099874	214807	215755
07111	099824	099807	099810	214379	215329
06888	099741	099714	099720	213778	214731
06666	099622	099584	099591	212949	213908
06444	099453	099401	099413	211827	212794
06222	099221	099153	099169	210333	211314
05999	098908	098818	098840	208382	209379
05777	098494	098375	098405	205879	206897
05555	097955	097798	097838	202725	203771
05333	097264	097060	097111	198825	199903
05111	096395	096130	096195	194087	195199
04888	095314	094977	095057	188438	189588
04666	093994	093570	093666	181822	183013
04444	092390	091880	091994	174215	175432
04222	090458	089881	090012	165623	166868
03999	088196	087551	087699	156092	157361
03777	085588	084874	085037	145704	146962
03555	082621	081841	082019	134579	135835
03333	079293	078452	078640	122871	124117
03111	075605	074713	074908	110761	111928
02888	071569	070637	070836	098450	099573
02666	067200	066244	066443	086147	087183
02444	062524	061563	061758	074066	074966
02222	057565	056622	056808	062411	063274
01999	052339	051455	051630	051374	051536
01777	045546	044732	044887	038712	039737
01480	039421	038696	038832	028913	029649
01270	033968	033340	033457	021443	022085
01087	029160	028617	028717	015781	016285
00928	024951	024483	024568	011551	011883
00790	021288	020858	020931	008393	008715
00670	018052	017704	017766	006037	006337
00566	015245	014953	015005	004316	004510
00475	012803	012556	012599	003045	003183
00396	010676	010472	010508	002116	002250
00327	008827	008661	008691	001458	001577
00267	007219	007080	007105	000988	001074
00216	005820	005708	005728	000649	000701
00170	004604	004514	004529	000409	000436
00131	003546	003476	003488	000243	000258
00097	002626	002574	002583	000134	000146
00067	001826	001790	001796	000065	000079
00041	001130	001108	001112	000025	000040
00019	000525	000515	000517	000005	000016
0	0	0	0	0	0

Table 3

Newton Algorithm Convergence Summary

Iteration Index (p)	Maximum Residual $\{\delta U\}_{j+1}^{p+1}$			
	2DBL Solution		2DPNS Solution	
	Blasius {V}	Computed {V}	Original	Revised
1	-1.2(-2)	-1.2(-2)	-1.2(-2)	-1.2(-2)
2	1.4(-4)	5.4(-5)	1.4(-4)	1.5(-5)
3	5.8(-7)	1.3(-5)	-7.6(-5)	-9.4(-5)
4		3.5(-6)	4.2(-5)	-5.3(-5)
5			-2.3(-5)	1.8(-5)
6			1.1(-5)	4.7(-5)
7			8.4(-6)	5.1(-5)
8				3.2(-5)
9				5.7(-6)

for the 2DBL $\{V\}_{j+1}^{p+1}$, column 3, the convergence is quadratic only to E(-4) for the two equation system. This decline in convergence rate for $|\delta U|_{max} < 10^{-4}$ characterizes all the 2DPNS algorithm solution constructions as well. The fourth column in Table 3 summarizes convergence of the original PNS algorithm (diagonal [JQQ], and solution for $\{P\}_{j+1}^{p+1}$ and $\{\phi\}_{p+1}^{p+1}$ using $\{QI\}_{j+1}^{p+1}$, which is monotonic and approximately linear for $|\delta U| < 10^{-4}$. The last column gives these data for the new algorithm construction, equation 64, with [JPU] and [JϕU] omitted (since they cause instability and eventual divergence). Convergence is quadratic to E(-4), and thereafter is nonmonotonic and sublinear (although these are extremum residuals occurring at different nodal coordinates at any iteration).

The current practice in CMC:3DPNS is 4-5 iterations per step with convergence set at $|\delta Q|_{max} \leq 10^{-4}$. Over this range the convergence character of the old diagonal [JQQ] construction and the coupled construction is nominally identical. Additional examinations were conducted to assess reasons for the poor convergence rate below E(-4). Since the action of the penalty term is applied modulo a discrete (second-order) approximation to $\partial/\partial y$, the {V} solutions generated at ten iterations/step or more will eventually exhibit "wiggles." The current practice is to element average either {δV} or {V} when this occurs. In Table 4, column 2 summarizes the standard test solution for {V} using the original algorithm (for reference), column 3 contains the revised algorithm data using {V} averaging, and column 4 contains the same algorithm solution without {V} or {δV} averaging. Close examination of the data in column 4 verifies periodic occurrences of local flat spots which will eventually grow into a $2\Delta y$ wave. Since a {V} and/or {δV} average is tantamount to not using the coupled implicit solution vector {δQI} as computed, this operation could contribute to poor convergence. A numerical test verified this to a limited extent, compare column 2 of Table 5 to column 5 of Table 3.

Alternatively, analysis determined that the "wiggles" in {V} can be traced back directly to {ϕ}, hence {Sϕ}, which if smoothed prior to use in equation 56 would not generate the discrete wave. The correct way to obliterate a $2\Delta y$ wave is to use a Shuman-type digital filter, cf., [12, Ch. 4]. As an approximation, $\{S\phi\}_{j+1}^{p+1}$ was simply averaged. The accuracy of the resultant solution for {V}, column 5 in Table 4, is nominally identical to the {V}-averaged solution data, except directly adjacent to the wall where the averaging removed the critical sensitivity. This operation did improve the iteration convergence, column 3 in Table 5, mostly in returning it towards monotonicity below $|\delta U|_{max} < 10^{-4}$.

The Newton Jacobian for these tests remained incomplete, since including either [JPU] or [JϕU] would destabilize the algorithm. (Both these terms involve a discrete approximation to an axial derivative {QI}′, see equations 62-63, to which the PNS penalty algorithm is quite sensitive.) The semi-implicit evaluation of {QI}′, of the form,

Table 4

2DPNS Algorithm Accuracy Comparisons, Laminar Test Case

NODES $2y/10\delta_o$	JUU,JVV PPRESS Q'Imp.	J(I,J) V Avg. Q'Imp.	J(I,J) No V Avg. Q' Imp.	J(I,J) Sϕ Avg. Q' Imp.	J(I,J) Sϕ Avg. Q' Sem-Imp.
07999	216400	214784	220248	218018	220856
07777	216260	214629	220080	217866	220733
07555	216054	214402	219993	217643	220549
07333	215755	214076	219559	217321	220280
07111	215329	213613	219416	216865	219893
06888	214731	212970	218562	216229	219347
06666	213908	212090	218138	215356	218586
06444	212794	210907	216796	214180	217550
06222	211314	209346	215631	212625	216159
05999	209379	207320	213703	210605	214330
05777	206897	204738	211318	208022	211964
05555	203771	201506	208441	204779	208961
05333	199903	197531	204574	200785	205219
05111	195199	192727	200052	195950	200644
04888	189588	187026	194729	190213	195140
04666	183013	180377	187884	183443	188642
04444	175432	172760	180839	175760	181157
04222	166868	164184	172174	167138	172609
03999	157361	154695	162172	157274	162973
03777	146962	144378	152939	147017	152646
03555	135835	133351	140295	135736	141402
03333	124117	121764	129053	123684	128928
03111	111928	109794	117596	111736	117288
02888	099573	097631	102326	099510	104491
02666	087183	085485	092874	086025	089834
02444	074966	073566	078062	075709	081177
02222	063274	061926	065551	061870	064422
01999	051536	050403	057151	052053	055122
01722	039737	039301	038727	039070	042943
01480	029649	029607	033067	028164	028567
01270	022085	021952	022816	022300	024686
01087	016285	016171	014982	015474	016477
00928	011883	011863	014978	011332	011714
00790	008715	008671	008086	008553	009200
00670	006337	006302	004866	006070	006634
00566	004510	004551	006336	004428	004565
00475	003183	003253	003469	003071	003123
00396	002250	002288	000324	002187	002038
00327	001577	001575	001338	001460	001242
00267	001074	001058	002293	001056	000790
00216	000701	000695	001278	000609	000633
00170	000436	000448	000155	000582	000575
00131	000258	000285	000164	000484	000406
00097	000146	000180	000691	000038	000072
00067	000079	000112	000843	-000280	-000305
00041	000040	000064	000503	-000419	-000526
00019	000016	000029	000098	-000325	-000435
0	0	0	0	0	0

TABLE 5

Newton Algorithm Convergence Summary

Revised 2DPNS Algorithm

	Maximum Residual $\{\delta U\}_{j+1}^{p+1}$			
			Semi-Implicit $\{Q\}'$	
Iteration Index (p)	Implicit $\{Q\}'$ No $\{V\}$ Avg.	Implicit $\{Q\}'$ $\{S\phi\}$ Avg.	$\{S\phi\}$ Avg. Full [J]	$\{S\phi\}$ Avg. Orig. [J]
1	-1.2(-2)	-1.2(-2)	-1.2(-2)	-1.2(-2)
2	-8.0(-5)	2.2(-5)	-8.2(-5)	9.4(-5)
3	-5.0(-5)	3.2(-5)	-3.3(-5)	-4.0(-5)
4	3.6(-5)	3.4(-5)	5.0(-5)	-2.8(-6)
5	4.1(-5)	2.2(-5)	4.1(-5)	
6	2.6(-5)	7.5(-6)	2.0(-5)	
7	1.4(-5)		8.7(-6)	
8	-1.1(-5)			
9	-1.3(-5)			

$$\{QI\}'_{j+1} \equiv \frac{1}{\Delta x_1}\left(\{QI\}^2_{j+1} - \{QI\}_j\right) = \frac{1}{\Delta x_1}\{\delta QI\}^2_{j+1} \qquad (66)$$

is the standard procedure in CMC:3DPNS. With this simplification, the parameter "a" in equations 62 and 63 is zero. As a consequence, [JϕU] vanishes identically and [JPU] can be included without destabilization. Using $\{S\phi\}$ averaging, the accuracy of this algorithm form is nominally unchanged, see column 6 in Table 4. The resultant Newton Jacobian is exact and these results are a very modest improvement in monotonicity of convergence, see column 4 of Table 5. This convergence character is the closest to the original algorithm, recall column 4, Table 3. Of more significance, insertion of $\{S\phi\}$ averaging into the original algorithm construction significantly improves convergence, see column 5 of Table 5. This operation yields the PNS algorithm as efficient as the direct 2DBL solution, recall column 3 of Table 3.

SUMMARY AND CONCLUSIONS

A generalized coordinates form of the penalty finite element algorithm for the 3-dimensional parabolic Navier-Stokes equations for turbulent subsonic flows has been derived. This algorithm formulation requires only three distinct hypermatrices in its formulation and is applicable using any boundary fitted coordinate transformation procedure. The tensor matrix product approximation to the Jacobian of the Newton linear algebra matrix statement has been derived. The Newton algorithm has been restructured to replace the large sparse matrix solution procedure with a grid sweeping procedure using α-block tridiagonal matrices where α equals the number of dependent variables.

The principal purpose of the reformulation is to improve solution economy. With the restructured Jacobian, solution economy is linearly dependent on the convergence (rate) of the Newton algorithm. A series of numerical experiments were performed to evaluate convergence as a function of Jacobian completeness and off-diagonal coupling. The results of these studies indicate that the favorable Newton quadratic convergence rate is maintained to a residual level of order 10^{-4}. Thereafter the convergence rate uniformly decreases to linear for residual computations in the range 10^{-4} - 10^{-5}. Several modifications to the implicitness of the algorithm Jacobian and to the overall linear algebra statement were made and evaluated. Comparison to exact solutions indicates adequate accuracy is attainable for each of the modifications.

The results of this study provide required guidance on the appropriate form for the tensor product 3DPNS algorithm. The original form of the algorithm, employing a diagonal Jacobian, retarded evaluation of the Poisson equation solutions, in particular the ϕ solution, and $\{S\phi\}$ averaging for the penalty term yields the best Newton convergence performance and solution accuracy for the test case. The considerable effort in constructing and coding the off-diagonal Jacobian entries appears unrewarded, based upon these data, especially considering the added solution costs associated with a block versus scalar tridiagonal matrix. This indication gains considerable importance in the progression to turbulent and/or three-dimensional flows, wherein the Reynolds stress tensor will almost double the block size. Based on these data, it appears that the 3DPNS tensor product algorithm should employ a nominally diagonal tensor product Jacobian approximation for the initial-value variables, and should retard the Poisson equation solutions, which in themselves can use a scalar diagonal tensor product Jacobian approximation. The computation of Reynolds stresses would also employ a scalar diagonal form when using an algebraic model. This reconstruction of the 3DPNS algorithm would fit directly into the present CMC:3DPNS code. It is suggested that this should indeed by the next step.

APPENDIX

Finite Element Algorithm Standard Hypermatrices

1. One-Dimensional Finite Element Domains R_1^e, k=1:

$$\{A10\} = \tfrac{1}{2}\begin{Bmatrix}1\\1\end{Bmatrix}$$

$$[A200] = \frac{1}{6}\begin{bmatrix}2 & 1\\ 1 & 2\end{bmatrix}$$

$$[A201] = \tfrac{1}{2}\begin{bmatrix}-1 & 1\\ -1 & 1\end{bmatrix}$$

$$[A211] = \begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}$$

$$[A3000] = \frac{1}{12}\begin{bmatrix}\begin{pmatrix}3\\1\end{pmatrix} & \begin{pmatrix}1\\1\end{pmatrix}\\ \begin{pmatrix}1\\1\end{pmatrix} & \begin{pmatrix}1\\3\end{pmatrix}\end{bmatrix}$$

$$[A3100] = \frac{1}{6}\begin{bmatrix}\begin{pmatrix}-2\\2\end{pmatrix} & \begin{pmatrix}-1\\1\end{pmatrix}\\ \begin{pmatrix}-1\\1\end{pmatrix} & \begin{pmatrix}-2\\2\end{pmatrix}\end{bmatrix}$$

$$[A3001] = \frac{1}{6}\begin{bmatrix}-\begin{pmatrix}2\\1\end{pmatrix} & \begin{pmatrix}2\\1\end{pmatrix}\\ -\begin{pmatrix}1\\2\end{pmatrix} & \begin{pmatrix}1\\2\end{pmatrix}\end{bmatrix}$$

$$[A3011] = \tfrac{1}{2}\begin{bmatrix}\begin{pmatrix}1\\1\end{pmatrix} & -\begin{pmatrix}1\\1\end{pmatrix}\\ -\begin{pmatrix}1\\1\end{pmatrix} & \begin{pmatrix}1\\1\end{pmatrix}\end{bmatrix}$$

$$[A3010] = \frac{1}{6}\begin{bmatrix}-\begin{pmatrix}2\\1\end{pmatrix} & -\begin{pmatrix}1\\2\end{pmatrix}\\ \begin{pmatrix}2\\1\end{pmatrix} & \begin{pmatrix}1\\2\end{pmatrix}\end{bmatrix}$$

$$[A3110] = \tfrac{1}{2}\begin{bmatrix}\begin{pmatrix}1\\-1\end{pmatrix} & \begin{pmatrix}1\\-1\end{pmatrix}\\ \begin{pmatrix}-1\\1\end{pmatrix} & \begin{pmatrix}-1\\1\end{pmatrix}\end{bmatrix}$$

2. Two-Dimensional Quadrilateral Domain R_e^2, k = 1:

$$[B200] = \frac{1}{9}\begin{bmatrix} 4 & 2 & 1 & 2 \\ & 4 & 2 & 1 \\ & & 4 & 2 \\ (\text{sym}) & & & 4 \end{bmatrix}$$

$$[B3010] = \frac{1}{36}\begin{bmatrix} -6 & -3 & -1 & -2 \\ -3 & -6 & -2 & -1 \\ -1 & -2 & -2 & -1 \\ -2 & -1 & -1 & -2 \\ 6 & 3 & 1 & 2 \\ 3 & 6 & 2 & 1 \\ 1 & 2 & 2 & 1 \\ 2 & 1 & 1 & 2 \\ 2 & 1 & 1 & 2 \\ 1 & 2 & 2 & 1 \\ 1 & 2 & 6 & 3 \\ 2 & 1 & 3 & 6 \\ -2 & -1 & -1 & -2 \\ -1 & -2 & -2 & -1 \\ -1 & -2 & -6 & -3 \\ -2 & -1 & -3 & -6 \end{bmatrix} \qquad [B3020] = \frac{1}{36}\begin{bmatrix} -6 & -2 & -1 & -3 \\ -2 & -2 & -1 & -1 \\ -1 & -1 & -2 & -2 \\ -3 & -1 & -2 & -6 \\ -2 & -2 & -1 & -1 \\ -2 & -6 & -3 & -1 \\ -1 & -3 & -6 & -2 \\ -1 & -1 & -2 & -2 \\ 2 & 2 & 1 & 1 \\ 2 & 6 & 3 & 1 \\ 1 & 3 & 6 & 2 \\ 1 & 1 & 2 & 2 \\ 6 & 2 & 1 & 3 \\ 2 & 2 & 1 & 1 \\ 1 & 1 & 2 & 2 \\ 3 & 1 & 2 & 6 \end{bmatrix}$$

REFERENCES

1. Baker, A. J., and Orzechowski, J. A., "An Interaction Algorithm for Three-Dimensional Turbulent Subsonic Aerodynamic Juncture Region Flow," AIAA Journal, V. 21, No. 4, 1983, pp. 524-533.

2. Baker, A. J., "Why a Finite Element Algorithm for the Parabolic Navier-Stokes Equations," in T. Cebeci (ed.), Proceedings Second Sym. on Numerical and Physical Aspects of Aerodynamic Flows, Cal. St. Univ./ Long Beach, 1983.

3. Baker, A. J., Orzechowski, J. A., and Stungis, G. E., "Prediction of Secondary Vortex Flowfields Induced by Multiple Free Jets Issuing in Close Proximity," Technical Paper AIAA-83-0289, 1983.

4. Baker, A. J., and Orzechowski, J. A., "A Penalty Finite Element Method for Parabolic Flow Prediction," ASME, App. Mech. Div. AMD-Vol. 51, 1982, pp. 137-142.

5. Baker, A. J., Yu, J. C., Orzechowski, J. A., and Gatski, T. B., "Prediction and Measurement of Incompressible Turbulent Aerodynamic Trailing Edge Flows," AIAA Journal, V. 20, No. 1, 1982, pp. 51-59.

6. Baker, A. J., and Orzechowski, J. A., "A Continuity Constraint Finite Element Algorithm for Three-Dimensional Parabolic Flow Prediction," in Ghia, K. N., et.al. (eds.), Proceedings Joint ASME-AIAA Sym. on Computers in Flow Predictions and Experiments, ASME, 1981, p. 103-117.

7. Baker, A. J., and Orzechowski, J. A., "An Assessment of Factors Affecting Prediction of Near-Field Development of a Subsonic VSTOL Jet in Cross-Flow," with J. A. Orzechowski, U. S. Navy Report NADC-81177-60, 1982.

8. Baker, A. J., "The CMC:3DPNS Computer Program for Prediction of Three-Dimensional, Subsonic, Turbulent Aerodynamic Juncture Region Flow. Vol. 1-Theoretical," NASA CR-3645, 1982.

9. Manhardt, P. D., "The CMC:3DPNS Computer Program for Prediction of Three-Dimensional, Subsonic, Turbulent Aerodynamic Juncture Region Flow, Volume II-User's Guide," NASA CR-165997, 1982.

10. Orzechowski, J. A., "The CMC:3DPNS Computer Program for Prediction of Three-Dimensional, Subsonic Turbulent Aerodynamic Juncture Region Flow, Volume III-Programmer's Manual," NASA CR-165998, 1982.

11. Cebeci, T., and Smith, A. M. O, Analysis of Turbulent Boundary Layers, Academic Press, New York, 1974.

12. Baker, A. J., Finite Element Computational Fluid Mechanics, McGraw-Hill/ Hemisphere, New York, 1983.

13. Thompson, J. F. (ed.), Numerical Grid Generation, North Holland Press, Amsterdam, 1982.

14. Halmos, P. R., Finite Dimensional Vector Spaces, D. Van Nostrand, Princeton, New Jersey, 1958.

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-172256		

4. Title and Subtitle	5. Report Date
PROGRESS ON A GENERALIZED COORDINATES TENSOR PRODUCT FINITE ELEMENT 3DPNS ALGORITHM FOR SUBSONIC FLOW.	December 1983
	6. Performing Organization Code: COMCO: 83 TR-2.1
7. Author(s): A.J. Baker and J.A. Orzechowski	8. Performing Organization Report No.
9. Performing Organization Name and Address: COMPUTATIONAL MECHANICS CONSULTANTS, INC. 3601A Chapman Highway Knoxville, Tennessee 37920	10. Work Unit No.
	11. Contract or Grant No.: NAS1-15105-MOD 6
12. Sponsoring Agency Name and Address: NATIONAL AERONAUTICS & SPACE ADMINISTRATION Washington, D.C. 20546	13. Type of Report and Period Covered: Contractor Report
	14. Sponsoring Agency Code

15. Supplementary Notes

LANGLEY TECHNICAL MONITOR: Douglas L. Dwoyer

16. Abstract

A generalized coordinates form of the penalty finite element algorithm for the 3-dimensional parabolic Navier-Stokes equations for turbulent subsonic flows has been derived. This algorithm formulation requires only three distinct hypermatrices and is applicable using any boundary fitted coordinate transformation procedure. The tensor matrix product approximation to the Jacobian of the Newton linear algebra matrix statement has been derived. The Newton algorithm has been restructured to replace large sparse matrix solution procedures with grid sweeping using α-block tridiagonal matrices, where α equals the number of dependent variables. Numerical experiments have been conducted and the resultant data gives guidance on potentially preferred tensor product constructions for the penalty finite element 3DPNS algorithm.

17. Key Words (Selected by Author(s))	18. Distribution Statement
Parabolic Navier-Stokes Equations Finite Element Penalty Algorithm Generalized Coordinates Tensor Matrix Products	Unclassified - Unlimited Subject Category 34

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	38	A03

For sale by the National Technical Information Service, Springfield, Virginia 22161